AMENDMENTS TO THE CLAIMS

Please add claims 22 and 23, as follows;

---

1.   (Previously Presented) A method for processing an exception in an emulator program running on a digital computer having a memory and under control of an operating system, the emulator program emulating execution of a user program constructed for execution on a legacy platform, the method comprising the steps of:

receiving an exception from the operating system during operation of the emulator program;

determining whether the received exception was caused by the emulator program itself or by the user program; and

if the exception was caused by the emulator program, handling the exception internally in the emulator program without delivering the exception to the emulated user program.

2.   (Original) A method for processing an exception according to claim 1 and further comprising, if the exception was caused by the user program:

identifying the type of exception;

determining whether the identified type of exception is currently blocked by the user program; and

if the identified type of exception is not currently blocked by the user program, delivering notification of the exception to the user program.

3   (Original) A method for processing an exception according to claim 2 and further comprising, if the identified type of exception is currently blocked by the user program, withholding delivery of the exception from the user program, and marking the exception as deferred for subsequent processing.

4 - 7. (Canceled)

8.   (Original) A method for processing an exception according to claim 3 further comprising creating and maintaining a status mask for indicating a

status as deferred or not deferred for each one of at least one predetermined type of exception; and wherein said marking the exception as deferred includes updating the status mask to indicate a status of the exception as deferred.

9.   (Original) A method according to claim 8 wherein the status mask is implemented as a predetermined data structure within the emulator memory space.

10.   (Previously Presented) A method for processing an exception according to claim 2 and further comprising, if the exception is determined to have been caused by the user program, and if the exception is not currently blocked by the user program,

determining whether the exception is synchronous or asynchronous; and if the exception is synchronous, delivering the exception to the user program.

11.   (Original) A method for processing an exception according to claim 10 and further comprising, if the exception is asynchronous, determining whether the exception indicates an interrupted system call; and if not, marking the exception as pending.

12.   (Original) A method for processing an exception according to claim 11 and further comprising, if the exception is asynchronous, and if the exception indicates an interrupted system call, delivering the exception to the user program.

13 - 20 (Canceled)

21.   (Previously Presented) A method for processing an exception according to claim 1 in which the exception is drawn from a set of exceptions including a privilege fault exception and a floating point arithmetic exception.

22.   (New) A method for processing an exception in an emulator program running on a digital computer having a memory and under control of an

operating system, the emulator program emulating execution of a user program constructed for execution on a legacy platform, the method comprising:

receiving an exception from the operating system during operation of the emulator program;

determining whether the received exception was caused by the emulator program itself or by the user program;

if the exception was caused by the emulator program, handling the exception internally in the emulator program without delivering the exception to the emulated user program; and

if the exception was caused by the user program:

identifying the type of exception;

determining whether the identified type of exception is currently blocked by the user program by examining a virtual exception mask that is maintained by the emulator program for simulating a user program exception mask as if the user program were executing on the legacy platform; and

if the identified type of exception is not currently blocked by the user program, delivering notification of the exception to the user program.

23.   (New) A method for processing an exception in an emulator program running on a digital computer having a memory and under control of an operating system, the emulator program emulating execution of a user program constructed for execution on a legacy platform, the method comprising:

maintaining a virtual exception mask for simulating a user program exception mask as if the user program were executing on the legacy platform;

receiving an exception from the operating system during operation of the emulator program;

determining whether the received exception was caused by the emulator program itself or by the user program; and

if the exception was caused by the user program:

determining whether the identified type of exception is currently blocked by the user program by examining the virtual exception mask; and

if the identified type of exception is not currently blocked by the user program, delivering notification of the exception to the user program.